**Ch.5 M‐files: Script Files**

 So far in this guide all the commands were executed in the **Command Window**. Using the **Command Window** to execute a series of commands is not convenient and may be difficult or even impossible. The problem is that the commands in the **Command Window** cannot be saved and executed again. In addition, the **Command Window** is not interactive. This means that every time the **Enter** key is pressed only the last command is executed, and everything executed before is unchanged. If a change or a correction is needed in a command that was previously executed and the results of this command are used in commands that follow, all the commands have to be entered and executed again. A different way of executing commands with MATLAB is first to create a file with list of commands, save it, and the run the file. This kind of files are called M‐files (because the extension m is used when they are saved). MATLAB uses two types of M‐files: **script** and **function files**. Script and function files should be saved in the current folder (default), which is available when working in the MATLAB domain. In this chapter we shall deal with script files.

عند الضغط على مفتاح Enter ينفذ الإيعاز الأخير فقط وآل ما تم تنفيذه سابقا يبقى بدون تغيير وعند القيام بتغيير أو تصحيح في إيعاز منفذ مسبقاً وتستخدم نتائج هذا الإيعازلاحقاً فان جميع الإيعازات يجب إدخالها وتنفيذها مرة أخرى. لتجاوز هذه المصاعب يتم آتابة الإيعازات في ملف يدعى (M\_files) وخزنه ثم تنفيذه. ويوجد نوعين من هذه الملفات ملف نصي وملف دوال.

**5.1 Definition**

A **script file** is a sequence of MATLAB commands called a **program**.

الملف النصي هو سلسلة من ايعازات ماتلاب ويدعى أيضاً **برنامج**.

**5.2 Notes on Script Files**

* When a script file runs, MATLAB executes the commands in the order they are written just as if they were typed in the Command Window.
* When a script file has a command that generates an output, the output is displayed in the Command Window.
* Using a script file is convenient because it can be edited (corrected and/or changed) and executed many times.
* Script files can be typed and edited in any text editor and then pasted into the MATLAB editor.
* The content of a script file may be a program, data, or just a set of instructions that are created using the Edit/Debugger built into MATLAB.
* It is preferable to write the commands **clear** and /or **close all** in the beginning of script files to clear the memory and/or close all opened windows.

عندما ينفذ ملف نصي فان إيعازاته تنفذ بالتسلسل كما كتبت و عندما يكون للملف النصي مخرجات فإنها تظهر في نافذة الايعازات. استخدام الملف النصي مريح لقابليته للتعديل ( إضافة و/ أو تغيير) والتنفيذ عدة مرات كما انه يمكن كتابة الملف بأي معالج نصوص (وورد مثلاً ومن ثم نسخه ولصقه في محرر ماتلاب).

**5.3 Creating and Saving a Script File**

In MATLAB script files are created and edited in the **Editor /Debugger Window**. This window is opened from the Command Window. In the **File** menu, select **New**, and the select **M‐file**. An open **Editor/Debugger** Window is shown in Figure (5.2).



**Figure (5.1): Creating M‐file**

****

**ايقونة التنفيذ Run في الطبعات الجديدة من ماتلاب تكون بشكل مربع أبيض بحافته اليسرى سهم أخضر**

**تطبع الإيعازات في الملف النصي صفاً بعد صف. ألأسطر ترقم تلقائياً. يبدأ صف (سطر) جديد بمجرد الضغط على مفتاحEnter**

**رقم الصف (السطر)**

**Figure (5.2): Editor/Debugger Window**

Once the window is open, the commands of the script file are typed line by line. MATLAB automatically numbers a new line every time the **Enter** key is pressed. The commands can also be typed in any text editor or word processor program and then copied and pasted in the Editor/Debugger Window.

Before a script file can be executed it has to be saved. This is done by choosing **Save** or **Save as**…from the **File** menu, selecting a location, and entering a name for the file. When saved (its icon will look like this one, MATLAB adds the extension .m to the file name. The rules for naming a script file follows the rules of naming a variable (see 1.8.1 pp.7). However, in this case a single dot is allowed within the name of the file; everything after the dot is used to tell MATLAB what type of file it is dealing with (whether it is a file containing MATLAB code, or data …etc). Avoid using one or two letters to name a file as it will conflict with variable names.

**5.4 Running (Executing) a Script File**

A script file can be execute either by typing its name in the Command Window and then pressing the Enter key, or directly from the Editor Window by clicking on the **Run** icon (see figure (5.2)). The file will be executed if the directory where the file is saved is the current directory of MATLAB or if the directory is listed in the search path.

**5.5 Current Directory**

The current directory is shown in the “Current Directory” field in the desktop toolbar of the Command Window, as shown in figure (5.3). The current directory can be changed either from the Current Directory field in the desktop toolbar:



**The current directory is**

**shown here**

**هنا يظهر الدليل الحالي**

**The current directory can be changed from here.**

**يمكن تغيير الدليل الحالي من هنا والضغط على المستطيل واختيار الدليل** **المطلوب كما يمكن التنقل بين الأدلة المستخدمة** **بالضغط على السهم الأسود**

**Figure (5.3): Current directory**

Or from the Current Directory Window (Figure (5.4)):

****

**Figure (5.4): Current Directory Window**

Or simply by typing the cd command in the Command Window (see laboratory guide). For Example, to change the current directory (C:\MATLAB6p5\work) to the folder “My Examples” in drive D, simply we write: >> cd('E:\My Examples').

Sometimes a window appears asking the user to change directory as follows:



**5.6 Search Path**

When MATLAB is asked to run a script file, or to execute a function, it searches for the file in directories that are listed in the search path. The directories that are included in the search path are displayed in the Set Path Window that can be opened by selecting **Set Path** in the **File** menu. Once the Set Path Window is opened, new folders can be added to, or removed from the search path.



**5.7 Input to a Script File**

When a script file is executed the variables that are used in the calculations within the file must have assigned values. The assignment of a value to a variable can be done in three ways, depending on where and how the variable is defined.

**5.7.1 The Variable is Defined and Assigned Value in the Script File**

In this case the assignment of value to the variable is part of the script file. If the user wants to run the file with a different variable value, the file must be edited and the assignment of the variable changed. Then, after the file is saved, it can be executed again. The following is an example of such a case:

**Example 5.1:**

Write a program in a script file to calculate the average of three numbers 45, 75, and 66 and then execute the program.

**Solution:**

The script file (saved as threenosaverage1.m)

% This script file calculates the average of three numbers

**ملاحظات**

% The assignment of the values of the umbers is part of the script file

clear

**إدخال المتغيرات ضمن الملف النصي**

num1=45;num2=75;num3=66;

no\_average=(num1+num2+num3)/3

The Command Window when this file is executed looks like:

>> threenosaverage1

**m ( تنفيذ الملف النصي بطباعة اسمه بدون اللاحقة (الامتداد**

no\_average =

**المتغير no\_average مع قيمته يظهر في نافذة الايعازات**

62

The program window is:



Another way to solve the example is by writing the following script file (saved as:

threenosaveragevect.m):

% This script file calculates the average of three numbers

**ملاحظات**

% The assignment of the values of the umbers is part of the script file

% The numbers are entered as elements of a vector

clear

num(1)=45;num(2)=75;num(3)=66;

**إدخال المتغيرات ضمن الملف النصي**

no\_average=mean(num)

The Command Window when this file is executed looks like:

>> threenosaveragevect

**تنفيذ الملف النصي بطباعة اسمه**

no\_average =

**المتغيرno\_average مع قيمته يظهر في نافذة الايعازات**

62

The deference between the first and the second method is that variables are entered in the first way singly, but in the second way as elements of a vector. Finding average in the second way is easier than the first way particularly for large number of variables. Average in the second way can be simply calculated by applying the array function **mean** instead of adding numbers and then divide the sum by their number.

**5.7.2 The Variable is Defined and Assigned Value in the Command Window**

In this case the assignment of a value to a variable is done in the Command Window. (

Recall that the variable is recognized in the script file). If the user wants to run the script

file with a different value for the variable, the new value is assigned in the Command

Window and the file is executed again.

**Example 5.2**

We shall use example 5.1, the script file (saved as threenosaverage2.m) is:

% This script file calculates the average of three numbers

% The assignment of the values of the umberts

% is done in the Command Window

no\_average=(num1+num2+num3)/3

**ملاحظة:** رفعنا عبارة **clear** لأنها ستؤدي إلى حذف القيم المدخلة من الذاآرة فيجب الانتباه إلى هذه العبارة.

The Command Window for running this file is:

>> num1=45;

**إدخال المتغيرات ضمن شاشة الإيعازات**

>> num2=75;num3=66;

**تنفيذ الملف النصي بطباعة اسمه**

>> threenosaverage2

no\_average =

**المتغير no\_average مع قيمته يظهر في نافذة الايعازات**

62

**5.7.3 The Variable is Defined in the Script File, but a Specific Value is Entered in the Command Window when the Script File is Executed**

In this case the variable is defined in the script file and when the file is executed, the user is prompted to assign a value to the variable in the Command Window. This is done by using the **input** command to create the variable. The form of the **input** command is:

**variable\_name=input ( ‘ string with a message that is displayed in the Command Window ‘ )**

When the **input** command is executed as the script file runs, the string is displayed in the Command Window. The string is a message prompting the user to enter a value that is assigned to the variable. The user types the value and presses the **Enter** key. This assigns the value to the variable. As with any variable, the variable and its assigned value will be displayed in the Command Window unless a semicolon is typed at the end of the **input** command.

**Example 5.3:**

We shall use example 5.1, the script file (saved as threenosaverage3.m) is:

% This script file calculates the average of three numbers

**ملاحظات**

% The assignment of the values of the numbers

% is done using the input command

clear

num1=input('The first number is: ');

**استخدام الايعاز input لادخال المعلومات**

num2=input('The second number is: ');

num3=input('The third number is: ');

**معادلة حساب المعدل**

no\_average=(num1+num2+num3)/3

The Command Window for running this file is:

>> threenosaverage3

**تظهر لنا العبارات المطبوعة في إيعاز input فندخل قيم الأعداد ونضغط على مفتاح Enter**

**تنفيذ الملف النصي بطباعة اسمه**

The first number is: 45

The second number is: 75

The third number is: 66

no\_average =

62 

In the previous examples scalars are assigned to the variables. In general, however, arrays (vectors and matrices) can also be assigned.

The **input** command can also be used to assign a string to a variable. This can be done in one of two ways. One way is to use the command in the same as shown above, and when the prompt message appears the string is typed in between two single quotes in the same way that a string is assigned to a variable without the **input** command. The second way is to use an option in the **input** command that defines the characters that are entered as a string. The form of the command is:

**variable\_name= input(‘prompt message’ , ‘s’)**

Where the ‘s’ inside the command defines the characters that will be entered as a string. In this case when the prompt message appears, the text is typed in without the single quotes, but it is assigned to the variable as a string.

**5.8 Echoing Commands**

The commands in a script M‐file will not automatically be displayed in the Command Window. If you want the commands to be displayed along with the results, use **echo** command: the **echo** command will show commands after it to the end, while the commands **echo on … echo off** will show commands between them on the Command Window.

**Example 5.4:**

Consider example 5.1, we can rewrite the script file to show the assignment command follows:

% This script file calculates the average of three numbers

% The assignment of the values of the numbers is part of the script file

clear

**اضافة الايعازين echo on وecho off يظهر العبارة التي تكتب بينهما**

echo on

num1=45;num2=75;num3=66;

echo off

no\_average=(num1+num2+num3)/3

The Command Window when this file is executed (saved as threenosaverage1echo) looks

like:

>> threenosaverage1echo

num1=45;num2=75;num3=66;

**ظهرت عبارة إسناد المتغيرات بسبب إضافة إيعازecho الى الملف النصي**

echo off

no\_average =

62

**5.9 Output Commands**

MATLAB has several commands that can be used to generate displays. The displays can be messages that provide information, numerical data, and plots. Three commands are frequently used to generate output are the **disp**, **display**, and **fprintf**. The commands can be used in the Command Window, in a script file, and in a function file. When these commands are used in a script file, the display output that they generate is displayed in the Command Window.

**5.9.1 The disp and display Commands**

The **disp** and **display** commands are used to display the elements of a variable and to display text on the screen. The **dis**p command doesn’t display the name of the variable, while the **display** command displays the name of the variable in addition to its value. The format of these commands is:

disp(‘text as string’)

disp(name of a variable)

 or

display(‘text as string’)

display(name of a variable)

 or

Every time the **disp** and **display** commands are executed, the display they generate appears in a new line.

**Example 5.5:**

>> a=[2 4 3;5 7 1];

>> disp(a)

**استخدام الايعاز disp يعرض المصفوفة a ولكن لا يعرض اسمها**

2 4 3

5 7 1

>> display('a 2 x 3 matrix')

**استخدام الايعاز display لعرض نص ونلاحظ هنا عرض الاسم للنص وبما اننا لم نسمي النص فيعطى له الاسم الافتراضيans**

ans =

a 2 x 3 matrix

**Example 5.6:**

We will write the script file of example 5.3 using the disp command (saved as threenosaverage4.m) as follows:



The Command Window for running this file is:



* The **disp** command is preferable than the **display** command since it doesn’t display variables names.
* Only one variable can be displayed in a **disp** or **display** command. If elements of two variables need to be displayed together, a new variable (that contains the elements to be displayed) must first be defined and then displayed.
* The following example shows that we can use the command **disp** to display output in a table.

**Example 5.7:**

The following script file shows how to display a population data in a table where population had been taken in years: 1984, 1986, 1988, 1990, 1992, 1994, and 1996. The population was 127, 130, 136, 145, 158, 178, and 211 million people.



When this script file (save as population\_table) is executed the display in the Command Window is:



يمكن كتابة الجدول بشكل آخر بتعريف أعمدته واحداً بعد الآخر أي الاستعاضة عن العبارةT=[yr’ pop’] في الملف population\_table بالعبارتين T(:,1)=yr’; T(:,2)=pop’;.

**5.9.2 The fprintf Command**

The **fprintf** command can be used to display output (text and data) on the screen or to save it to a file. With this command (unlike the **disp** command) the output can be formatted. The **fprintf** command has many available options.

**5.9.2.1 Using the fprintf Command to display text**

To display text, the **fprintf** command has the form:

**fprintf( ‘ text typed in as a string ‘ )**

**Example 5.8:**

fprintf( ‘ My name is ABC,I am a student in the second class. ‘ )

If the previous line is a part of a script file, when the line is executed, the following is

displayed on the Command Window:

My name is ABC,I am a student in the second class.

* With fprintf command, there are several characters, called the escape characters. They are used to control the display. The escape characters are:

|  |  |  |  |
| --- | --- | --- | --- |
| Character | Description | Character | Description |
| \b | Backspace | \t | Horizontal tab |
| \f | Form feed | \\ | Backslash |
| \n | New line | \”or” | Single quotation mark |
| \r | Carriage return | %% | Percent character |

 **Table (5.1): The Escape characters**

**Remark:** These characters can be used also in the input command.

**Examples:**

Using the line: fprintf( 'Solution is not found ')



* When a program has more than one **fprintf** command, the display that they generate is continuous (the **fprintf** command does not automatically start a new line). This is true even if there are other commands between the **fprintf** commands. To start a new line with the **fprintf** command, a \n must be typed at the start of the string.

**Example 5.19:**

Consider the following script file:

fprintf(' The problem, as entered, has no solution. Please check the input data.')

x=6 , d=19+x ;

fprintf('Try to run the progrm later.')

y=x\*d ;

fprintf('Use different input values.')

When this file is executed (using **run**), the following is displayed on the Command Window:

>> The problem, as entered, has no solution. Please check the input data.x =

**يلاحظ تتابع طباعة محتويات إيعازاتfprintf**

**على الرغم من وجود عبارات بينها**

6

Try to run the progrm later.Use different input values.

**5.9.2.2 Using the fprintf Command to Display a Mix of Text and Numerical Data**

To display a mix of text and number (value of a variable), the fprintf command has the form:

**fprintf ( ‘ text as string % -5.2 f additional text ‘, variable\_name )**



The formatting elements are:



The field width and precision (5.2 in the previous example) are optional. The first number (5 in the example) is the field width that specifies the minimum number of digits in the display. If the number to be displayed is shorter than field width, spaces or zeros are added in front of the number. The precision is the second number (2 in the previous example) specifies the number of digits to be displayed to the right of the decimal point.

Conversion characters specify the notation of the output. Some of the common notations are:



**Example 5.20:**

Various examples to display the variable x=1234.5671 are written in a script file:





* With the fprintf command it is possible to insert more than one number within the text. This is done by typing %g (or % followed by any formatting elements) at the places inthe text where the numbers are to be inserted. The after the string argument of the command (following the comma), the names of the variables are typed in the order that they are inserted in the text. In general the command looks like:

fprintf( ‘… text … % g … % g … % f … ’, variable1,variable2,variable3)



When this script file (saved as Example5\_21) is executed, the display on the Command Window is:

>> Example5\_21

A projectile shot at 30.00 degrees with a velocity of 1584.00 km/h will travel a distance of 17.091 km.

* The fprintf command is vectorized. This means that when a variable that is a vector or a matrix is included in the command, the command repeats itself until all the elements are displayed. If the variable is a matrix the data is used column by column.

**Example 5.22**

The script file below creates a 2 x 5 matrix T in which the first row are the numbers 1 through 5, and the second row are the corresponding square roots.



When this script file (saved as Example5\_22) is executed, the display on the Command Window is:



**Example 5.24: A student degrees**

The student ABC has the degrees 50, 60, 64, 70, 75, 88, 57. Write a script file to print her degrees and her average.